Examination 1
COP 4600 Operating Systems
17 October 2000

Instructions

1. This is a closed-book examination.
2. You are permitted on 8.5 by 11 inch sheet of notes, both sides, that you have prepared.
3. You are permitted 60 minutes to complete this examination.
4. **Do not start** the exam until the proctor has told you to start.
5. **Answer any three (3) questions, and no more.** All questions are of equal value.
6. **Leave sufficient room in the upper lefthand corner for the staple.**
7. Use exactly one page of paper (both sides is OK) to hold the answer to each question, and please write legibly.
8. Start the answer to each question on a new page (i.e., do **not** put the answer to more than one question on the same page).
9. Assemble your answers in numerical order of the questions when you submit them,
10. Include your last name and page number in the upper right hand corner of each answer page.

**Read and sign the following statement.** You may write this on your exam and sign it there if you wish to take the exam questions home with you today.

**On my honor, I have neither given nor received unauthorized aid on this examination.**

Signed:
1. (a) Describe in detail what happens as a typical personal computer is powered on, as it relates to
the operating system. Include all code that is executed, where the code is when the machine
was started, and what each piece of code must do.
(b) What is the difference between an interrupt, an interrupt vector, and an interrupt handler?
(c) How do an operating system and its underlying hardware manage interrupts? What does an
interrupt handler typically do?
(d) What is the difference between major and minor device numbers?

2. Consider a system in which there are four processes \((P_1, P_2, P_3, P_4)\) and three resource types
\((T_A, T_B, T_C)\), with one instance of resource type \(T_A\) and two instance each of resource types \(T_B\)
and \(T_C\). Currently, \(P_1\) has been allocated one instance each of \(T_A\) and \(T_B\), \(P_2\) has been allocated
one instance of type \(T_B\), and \(P_3\) has been allocated an instance of \(T_C\). \(P_1\) requests an instance
of \(T_C\), \(P_2\) requests one instance each of \(T_A\) and \(T_C\), \(P_4\) requests one instance each of \(T_B\) and \(T_C\).

(a) Draw the Wait-For Graph for this system state.
(b) Draw the Resource Allocation Graph for this system state.
(c) Is the system currently in deadlock or not? Justify your answer.
(d) Let the maximum requests that can be made for processes be given as vectors, and let \(P_1\)’s
maximum request vector be \((1, 2, 1)\), then \(P_1\) may request an additional instance of resource
type \(T_B\). Let \(P_2\)’s maximum request vector be \((1, 1, 1)\), \(P_3\)’s maximum request vector be
\((0, 1, 1)\), and \(P_4\)’s maximum request vector be \((1, 1, 1)\). (So, for example, \(P_2\) has already made
its maximum request, but \(P_3\) has not). Is the current system state safe or not? Justify your
answer.

3. (a) Masking interrupts is a common way to protect a critical section. What limitations does this
approach have, and why? What must be done in order to implement this approach, and why?
(b) Suppose you have a multiprocessor with 1024 processors, and you have to program the system
to compute new values for each cell in a large grid based upon the previous values in it and the
adjacent cells in a discrete time simulation (e.g., new values are computed for each microsecond
of simulated time). Grid values are stored in shared memory. What method (if any) would you
use to protect the grid values and what method (if any) is needed to synchronize the
processing of grid elements? Explain.
(c) How would your answer change if you had one powerful processor that was simulating the
multiprocessor with 1024 processes?
4. (a) How may threads be used to provide asynchronous I/O or message passing when the system only supports synchronous calls? What constraints must there be on the types of threads that are used to do this? Why?

(b) Give two examples of systems in which it is critical that no process starve or experience deadlock. Justify your answers.

(c) Many systems do not attempt to avoid or detect deadlock. Why not?

<table>
<thead>
<tr>
<th>Process Number</th>
<th>Priority (1—high)</th>
<th>Arrival Time</th>
<th>Burst Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>0</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

5. Consider the following set of process CPU bursts.

(a) Give Gantt charts for this set for FCFS, SJF, SRTF, RR(2), RR(3) and preemptive Priority. Show the ready queue at "interesting" times.

(b) What is the average turnaround time for each of those scheduling policies? Compare and explain.